Sentiment extraction from Consumer-generated noisy short texts

Hardik Meisheri
TCS Innovation Labs
New Delhi, India
hardik.meisheri@tcs.com

Kunal Ranjan
TCS Innovation Labs
New Delhi, India
k.ranjan@tcs.com

Lipika Dey
TCS Innovation Labs
New Delhi, India
lipika.dey@tcs.com

***Abstract*—Sentiment analysis or recognizing emotions from short and noisy text from social networks such as twitter has been a challenging task. Most of the existing models use word level embeddings for the final classification of the sentiments. This paper proposes a novel representation of short text derived from a combination of word embeddings and character embeddings using Bidirectional LSTM (BiLSTM). Along with this, we use attention mechanism that learns to focus on sentiment specific words. Robust representation of short text can be applied for sentiment classification as well as predicting intensity of sentiments. This paper presents evaluation of proposed model on classification as well as regression dataset. Results show significant improvement in results as compared to baselines of respective datasets.**

***Index Terms*—Deep learning, LSTM, Sentiment analysis.**

I. Introduction

In natural language processing, sentiment classification refers to the task of labeling a text as emanating positive or negative sentiment as a whole. On the other hand, emotion recognition is the task of associating words, phrases or documents with a set of predefined emotions from psychological models like fear, joy, anger, sadness, disgust, anxiety, surprise etc. The strength of the emotions expressed in text helps to quantify and compare subjective expressions and can be used downstream as well.

Twitter, with its large social reach, has become a rich source of user sentiments and emotions, with far reaching consequences in shaping the future of products, services or organizations. Due to limitations imposed on the maximum number of characters that can be used in a single tweet, users continuously devise novel and innovative ways to communicate often defying the traditional language syntax. Thus the resulting text can be highly ”noisy” for traditional Natural language processing algorithms and mechanisms that are designed to use Part of speech tags and dependency tags. In recent times, several deep learning models have been proposed for these tasks. While we shall be discussing some the important initiatives in the next section, it may be pointed out here that most of these use ’word’ based approaches. Our observation is that tweets specifically, and user-generated text on social media more commonly, often use innovative expressions using new or out of vocabulary words in combination with icons, numbers and symbols to express themselves. Word-based systems fail to learn these expressions effectively, thereby affecting the performance of emotion recognition or sentiment classification down the line.

In this paper, we propose a combination of word level representation to learn macro level features and character level representation to learn micro level features for different emotion classes, using Bidirectional LSTMs, initially proposed in [1]. In addition, we propose the use of attention mechanism to enhance the performance of the learning systems. We show that the proposed combination out-performs state of the art sentiment classification mechanisms. We also show that given specific emotion labels, the proposed network can predict the intensity of the emotion correctly by posing it as a regression problem.

II. Related Work

Traditional sentiment analyzers [2]–[4] that worked fine with well-written texts, face challenges at lexical, syntactic and semantic levels when dealing with tweets [5].The state-of-the-art approaches to sentiment analysis of social media data work on three point scale of sentiments using word embeddings. Two different word embeddings are used for this purpose - word2vec [6] and Glove [7].

In [8], learning of sentiment specific word embedding is presented. The embeddings were learnt from a large corpus using word2vec model, while convolutional Neural networks (CNN) followed by max-pooling [9]–[11] were used as classifier. In [12] distant-supervised method to learn set of embeddings using word2vec and glove, stacked layers of CNN and gradient boosting trees are used for classification.

Distant-supervised methods presented in [8], [12] have core dependency on massive data that needed to be mined based on emoticons. Positive and negative emotions can be misleading as presented in [13]. In addition, sarcasm present in tweets can altogether confuse a system, since emoticons or words used in sarcastic text do not correspond to the actual emotions expressed [3]. Further, these methods cannot take care of learning to predict the granularity of emotions, say for example in a range of 0 to 1.

In past literature, lexicon-based features have been used along with neural network models to predict intensity of emotions [14]–[21]. However these features are not helpful since they depend on domain-specific fixed dictionaries.

In this paper, we have proposed methods which can effectively detect the intensity of an emotion and classify sentiment to three or five point scale without depending on fixed vocabularies.

Rest of the paper is organized as follows, section III describes the proposed models and deep learning modules such as LSTM, CNN and BiLSTMs and the motivation behind choosing them. In section IV discussion on experimentation related to SemEval and WASSA dataset is presented. Finally section V concludes the paper.

III. Proposed Method

Text from twitter possesses a high degree of variation with respect to grammatical structure and syntax. As mentioned earlier twitter imposes the character limit of 140 to a single tweet. This limit excludes the user mentions, where user mention is defined as mentioning user-names of the user in the tweet. In addition to this, a emoticon is considered as a single character if it has been encoded in UTF format. Twitter text also contains hyperlinks to video and images that are posted along with a tweet. Not all these features are required for emotion analysis. A preprocessing of text is carried out to remove unnecessary elements before feeding it to the network. The details of this step are given later.

Datasets that are available for sentiment analysis are crowd-sourced and annotated by a large group of users. Since sentiment itself is a subjective matter, users may have different perception of sentiments for the same text. Besides, a users inherent beliefs or knowledge also play a role in sentiment perception. Table I shows sample tweets from SemEval-2016 dataset to illustrate these. Class represents the sentiment of that tweet, -2 being most negative, 2 being most positive and 0 is neutral. The first two tweets, though same have got different classes assigned by different users. The last two tweets are almost similar apart from the names of people, where Angela Merkel is replaced by Bobby Jindal, but their sentiment classes are different. This kind of dependency is very hard to capture from word embeddings that are pre-trained. It is observed that this kind of noise becomes more evident as the granularity of sentiment prediction increases.

Training our own embeddings is not possible for this kind of tasks as annotated data is in order of hundreds, which is very low for learning embeddings. We propose that these types of dependencies can be tackled by using character level embeddings.

Table II shows another set of tweets from WASSA dataset. In this dataset, a tweet is presented with an underlying emotion and its intensity. First two tweets from this dataset are also similar apart from a hashtag word that is present in the latter. Using of hashtags is common practice in social networks where user associate a sentence to one or more hashtags. The hashtags maybe combinations of one more words without any separation. The sentiment intensity associated to both of this tweet differ to a considerable extent just because of a presence of single hashtags. The third and fourth tweets in this table illustrate another unique nature of hashtags that are being used. They are combinations of words and digits which are not possible to be found in pre-trained embeddings.

Character level embeddings are necessary to create representations of this kind of out of vocabulary words. In addition to character level embeddings, we propose that dependencies of emotions on hashtags can be modeled using Attention mechanisms that have gained popularity in the field of computer vision.

Fifth and sixth tweets in the table are interesting examples, where we have same hashtags depression and anxiety yet we have a drastic difference in the intensity levels. The sequence of words and context also play a major role and relying on just hashtags is not sufficient. Last two tweets are on another side of a spectrum where an addition of hashtags does not cause any change in the resulting intensities. The proposed method to use combinations of word-level and character level embeddings along with attention mechanism can handle all these cases quite effectively. We now present the sequence of steps followed to process the tweet texts and learn classification models to predict sentiment or emotion intensities.

A. Preprocessing

As stated earlier text from tweets is inherently noisy. They contain twitter specific words along with hashtags and username mentions. For word and character level models, two sets of data are generated from raw data. For word level following cleaning is applied,

- **Hashtags** are important markers for determining sentiment or user intent often user mention popular hashtags to show their intent. The ”#” symbol is removed and the word itself is retained.
- **Username mentions**, i.e.words starting with ”@”, generally provide no information in terms of sentiment. Hence such terms are removed completely from the tweet. If however, the text contains multiple tweets as part of a single conversation, the user mentions would have been an important aspect.
- **Links** Hyperlinks are removed as they convey no information about the sentiment present in the text.
- **Emoticons** (for example, ‘:(’, ‘:)’, ‘:P’ etc) are removed.
- Extra spaces are removed.

While for character level model, user mentions and hashtag symbol is removed as described earlier, whereas emoticons are replaced with corresponding description of that emoticon, for e.g., *U+1F600* is replaced by word ‘face with smile’. This modification helps in capturing sentiments described by this emoticon in form of text.

B. Proposed Architecture

Proposed model is shown in Figure 1. There are two parallel processing layers which takes word and character as inputs to predict the intensity or class of sentiment. Text is preprocessed before feeding into the model as explained earlier. Details of each module is explained in following sections.

TABLE I
SAMPLE TWEETS FROM SEMEVAL-2016 DATASET

Tweets	Class
Did you know that 'Angela Merkel' was Trending Topic on Wednesday 2 for 6 hours in Sweden? trndnl	0
Did you know that 'Bobby Jindal' was Trending Topic on Friday 24 for 7 hours in Baton Rouge? trndnl	1
When you realize classes start the 9th so the AC/DC concert will be the best concert since Eminem and Jay-Z thatwasgoat	2
When you realize classes start the 9th so the AC/DC concert will be the best concert since Eminem and Jay-Z thatwasgoat	1

TABLE II
SAMPLE TWEETS FROM WASSA DATASET

Tweets	Emotion	Intensity
@huwellwell One chosen by the CLP members! MP seats are not for people to dole out to their mates, we elect candidates. #fuming	anger	0.682
@huwellwell One chosen by the CLP members! MP seats are not for people to dole out to their mates, we elect candidates.	anger	0.438
Having a terrific Tuesday? Crush it today with the Power of 4. Treat your internet like Pizza =D #PowerOf4	fear	0.250
I'm excited for the #FirstDayofFall & the rest of the season. I have 2 #Halloween #scare events I'm covering for @ThrillzCo in the next week	fear	0.188
Panic attacks are the worst. Feeling really sick and still shaking. I should be a sleep. #anxiety #depression	sadness	0.917
How brave are the young individuals that have opened up to us about their #depression and #anxiety to help raise awareness!! @beyondblue	sadness	0.479
This is not me brown nosing but I've listened to lots of housing ministers but @GavinBarwellMP #nhf16 impressed me more than any	joy	0.354
This is not me brown nosing but I've listened to lots of housing ministers but @GavinBarwellMP #nhf16 impressed me more than any #optimism	joy	0.354

C. Embeddings

Processed text from tweets is converted to fixed dimension vector using pre-trained embeddings available in the literature. In this paper we have used pre-trained GloVe Word Embeddings [7] which were trained over large corpus of tweets. Pre-trained embeddings of 25, 50, 100 and 200 dimensions are provided. For this work, we use the 200-dimensional vectors. In addition to this, character embeddings were used which were trained on 840 Billion tokens from common crawl dataset[1]. It provides 300 dimension vector for each character. There are total *94* characters included which contains all alphabets lower and upper case along with numbers and punctuations.

Tweet can be represented in form of matrix using word embeddings as $\langle n_w \times d_w \rangle$, where n_w is maximum number of words present in a tweet and d_w is the dimension of each word embedding. For this paper we have assumed $n_w = 40$, this assumption is in line with the 140 characters limit on each tweet. Each tweet is thus represented as a $\langle 40 \times 200 \rangle$ matrix. Zeros are appended in the beginning if number of words is less than n_w in a tweet.

Tweet can also be expressed as a sequence of characters with its corresponding dimension generating a matrix. Let us denote this matrix with dimension as $\langle n_c \times d_c \rangle$, where n_c is the maximum number of characters present in a tweet and d_c is the dimension of each character. As stated earlier a emoticon is replaced by text we assume n_c in a tweet to be 180. Therefore, each tweet can be represented as a $\langle 180 \times 300 \rangle$ character embedding matrix.

D. Bidirectional LSTM Recurrent Neural Networks

Recurrent networks have been effective in handling temporal data. Most commonly, recurrent neural networks (RNNs) are trained with stochastic gradient descent (SGD), where the gradient of the training criterion is computed with the backpropagation through time algorithm [22]–[24]. Long Short Term Memory networks (LSTM) [25] are special case of RNNs where they tackle the classical problem of vanishing (or exploding) gradients [26]. LSTM architecture that is used here is same as proposed in [27] which is governed by following equations,

$$\begin{gathered}
i_t = tanh(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \\
j_t = sigm(W_{xj}x_t + W_{hj}h_{t-1} + b_j) \\
f_t = sigm(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \\
o_t = tanh(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \\
c_t = c_{t-1} \oplus f_t + i_t \oplus j_t \\
h_t = tanh(c_t) \oplus o_t
\end{gathered}$$

In these equations, the W_* variables are the weight matrices and the b_* variables are the biases. The operation $\oplus$ denotes the element-wise vector product. The variable c_t denotes memory of LSTM at time step t. h_t is referred to as hidden state.

Bidirectional recurrent neural networks (BRNNs) were initially proposed in [1] for speech recognition task. Gradually they have been applied to different tasks like parsing [28] and spoken language understanding [29]. Bidirectional Long Short Term memory (BiLSTM) networks are BRNNs using LSTM hidden layers which were proposed in [30]. BiLSTMs are two LSTM stacked over each other. Forward pass processes the information from $t = 1 - T$ while, backward pass processes information from $t = T - 1$. Equations for the LSTM layers remain same and training can be done using stochastic gradient

[1] https://github.com/minimaxir/char-embeddings

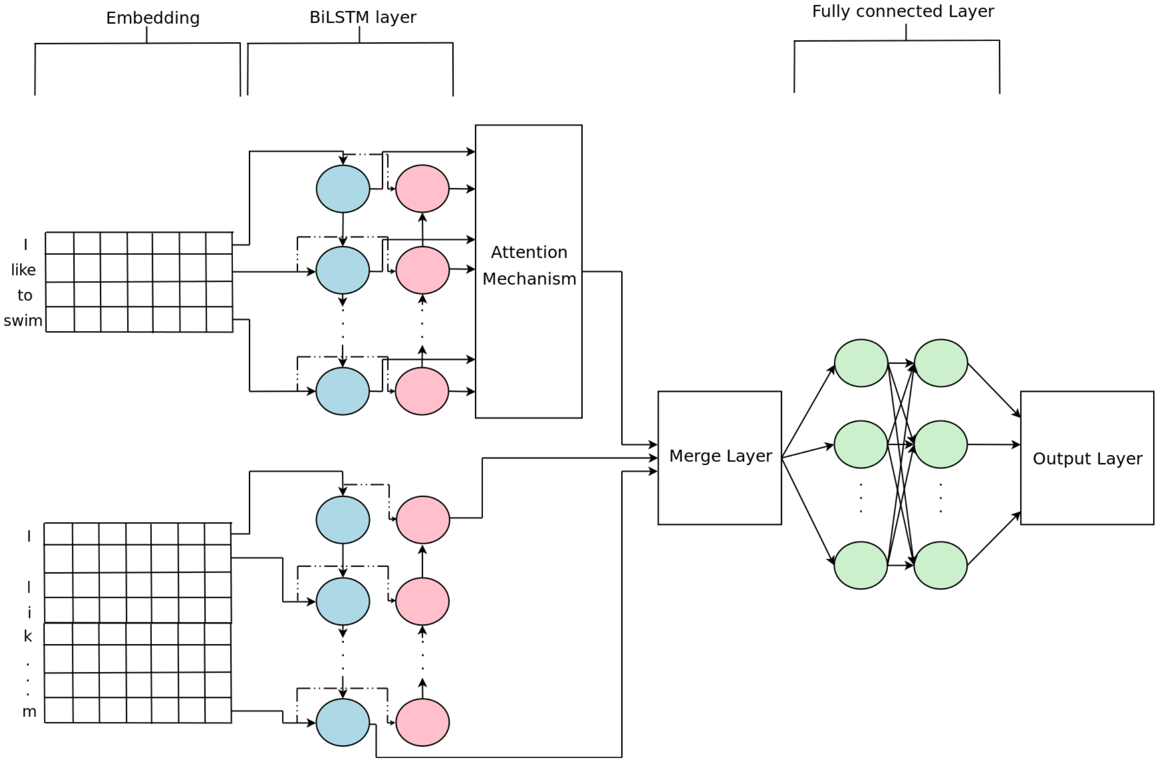

Fig. 1. Model_Architecture

descent as shown in Figure 2.

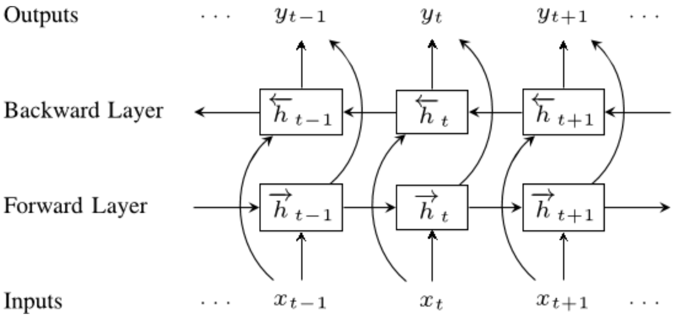
Outputs
$\cdots$ y_{t-1} y_t y_{t+1} $\cdots$
Backward Layer
$\overleftarrow{h}_{t-1}$ $\overleftarrow{h}_t$ $\overleftarrow{h}_{t+1}$
Forward Layer
$\overrightarrow{h}_{t-1}$ $\overrightarrow{h}_t$ $\overrightarrow{h}_{t+1}$
Inputs
$\cdots$ x_{t-1} x_t x_{t+1} $\cdots$

Fig. 2. Information flow in Bidirectional flow

In this paper we have utilized BiLSTM to effectively capture long term dependencies over word and character embeddings. We have used single layer of BiLSTM for both classification and regression task. In the above equation of LSTM's, forget gate bias is initialized to a random value which can prove to be inefficient as it can introduce problem of vanishing gradient by a factor of 0.5 [25], [31]. This problem affects the long term dependencies adversely which is critical to our task. For this we have initialized forget gate bias b_f to value just above 1. This enables the gradient flow in the network [32], [33]. In addition, dropout has been used to avoid over-fitting in BiLSTM layer. This increase in performance is significant as reported in the result section.

E. Attention Mechanism

Attention mechanism was recently proposed for modeling long-term dependencies. It allows for more direct relationships between state of the model at different point in the temporal domain. Following the definition from [34], given a model which produces a hidden state h_t at each time step, attention-based models compute a context vector c_t as the weighted mean of the state sequence h by

$$c_t = \sum_{j=1}^{T} \alpha_{tj} h_j \tag{1}$$

where T is defined as the total number of time steps in sequence and α_{tj} is a weight vector computed at each time step t for each hidden state h_t. These context vectors are then used to compute a new state sequence s, where s_t depends on s_{t-1}, c_t and the model's output at $t-1$. The weightings α_{tj} are then computed by,

$$e_{ij} = a(s_{t-1}, h_j) \tag{2}$$

$$\alpha_{tj} = \frac{exp(e_{tj})}{\sum_{k=1}^{T} exp(e_{tk})} \tag{3}$$

where a is function which can be thought of as computing a scalar importance value for h_j, given the value of h_j and the previous state s_{t-1}.

This formulation allows new state sequence s to have more direct access to the entire state sequence h. As a consequence in word context, output at each time-step of BiLSTM is given a weightage. This helps in learning to give more focus on words that are more significant to the task in hand, in our case sentiment. Output of this attention mechanism can be seen as a weighted multiplication of hidden states at each time step of BiLSTM.

F. Fully Connected Layers

We have used 3 layers of fully connected neurons including output layer. For classification, output layer contains 5 neurons as number of classes are five for SemEval-2016 dataset with softmax as activation function for probabilistic categorization. Whereas for regression task output layer contains 1 neuron and sigmoid as activation function for predicting intensity values between $0 - 1$. For remaining two fully connected layers, we have used Scaled Exponential Linear Units (selu) [35] as activation function. Selu was proposed recently and have shown improvements in performances by acting self normalizing layers and removes the need for batch normalization layers. It can be mathematically expressed as,

$$selu(x) = \lambda \begin{cases} x & if x > 0 \\ \alpha e^x - \alpha & if x \leq 0 \end{cases} \tag{4}$$

Hidden units of each layer for both the classification and regression task are shown in Table III

TABLE III
HIDDEN UNITS IN MODEL

Layers	*Classification*	*Regression*
BiLSTM Layer (word)	20	20
BiLSTM Layer (char)	60	70
Fully connected layer 1	40	40
Fully connected layer 2	10	7
Fully connected layer 3	5	1

IV. RESULTS AND DISCUSSION

A. Datasets

Proposed model has evaluated on two datasets namely, SemEval-2016 dataset and WASSA-2017 dataset. Both of these dataset concentrates on sentiment represented by a tweet. SemEval dataset is used as a classification task and WASSA dataset is used as a regression task. This datasets were annotated using crowd-sourcing.

1) SemEval-2016 Task 4 Subtask C: This dataset provides a tweet and its corresponding sentiment class. Classes are most negative, negative, neutral, positive and most positive denoted in range of -2 to 2 respectively. Sample tweets of this dataset are shown in Table I. Task can be summarized as *"Given a tweet known to be about a given topic, estimate the sentiment it conveys towards the topic on a five-point scale ranging from highly negative to highly positive."* Figure 3 shows histogram of the target class values, high sentiments are rare to find and that can be reflected in the histogram where most of the sentiments are in mild positive range.

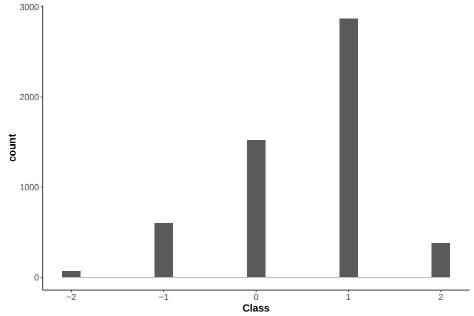

Fig. 3. Distribution of class values in SemEval-2016 dataset

2) WASSA-2017 Shared Task on Emotion Intensity: Dataset was taken from WASSA-2017 Shared Task on Emotion Intensity [36]. Task can be summarized as *"Given a tweet and its corresponding emotion, predict the intensity score of that emotion between 0 to 1, 0 being lowest and 1 being highest."*

Sample tweets are presented in Table II. Datasets contains tweets for four kinds of emotion, and during prediction of intensity type of emotion is provided. Training distribution of these 4 emotions are shown in Figure 4, Figure 5, Figure 6 and Figure 7. As observed earlier in this dataset also extreme emotions are rare and most of the intensity is concentrated from 0.4-0.8.

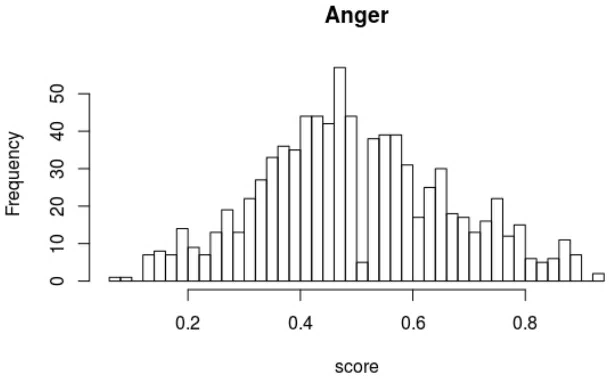

Fig. 4. Distribution of Intensity values for Anger Emotion

Data size and distribution over train, development and test set provided are presented in Table IV. For evaluation purposes in this paper, train and development set has been used for training the model and results are presented on test dataset.

TABLE IV
DETAILS OF DATASETS

Dataset	*Train*	*Development*	*Test*
Semeval-2016	5338	1784	20632
WASSA-Anger	857	84	760
WASSA-Fear	1147	110	995
WASSA-Joy	823	79	714
WASSA-Sadness	786	74	673

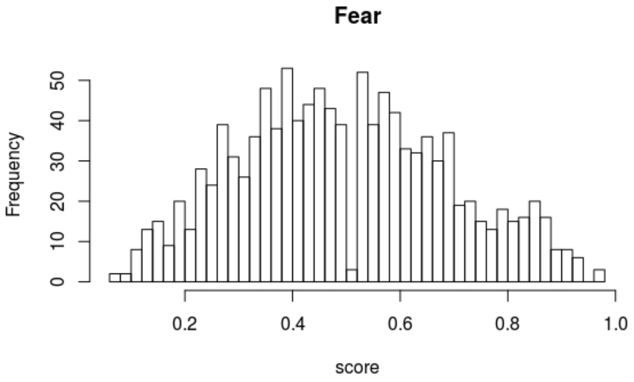

Fig. 5. Distribution of Intensity values for Fear Emotion

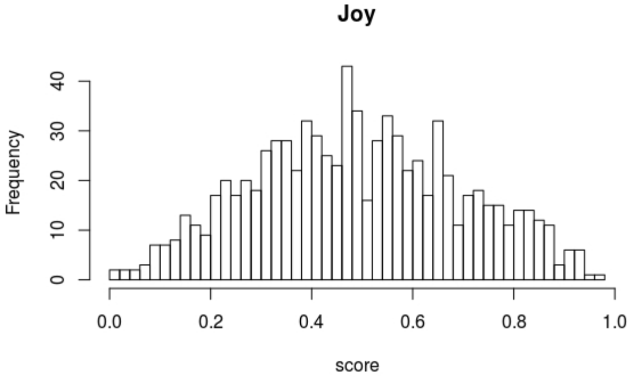

Fig. 6. Distribution of Intensity values for Joy Emotion

B. Evaluation Metrics

Results are reported for official evaluation of the datasets that used. For SemEval-2016 dataset two different metrics were used. Modified version of mean absolute error was used to evaluate models as there is a huge imbalance in the distribution of classes as explained earlier. The metrics are presented as follows,

$$MAE^M(h, Te) = \frac{1}{|C|}\sum_{j=1}^{|C|}\frac{1}{|Te_j|}\sum_{x_i \in Te_j}|h(x_i) - y_i| \quad (5)$$

where y_i denotes the true label of item x_i, $h(x_i)$ denotes its predicted label, Te_j denotes set of documents whose true class is c_j, $|h(x_i) - y_i|$ denotes the distance between classes $h(x_i)$ and y_i. MAE^M is a mean absolute error over each class.

For WASSA dataset pearson and spearman correlation were used as evaluation metrics, Pearson correlation coefficient can be defined as,

$$pearson = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \quad (6)$$

where x_i is the true value, y_i is predicted value, $\bar{x}$ and $\bar{y}$ are mean of true and predicted values respectively and n is total number of samples present. Spearman correlation coefficient is defined as pearson correlation of ranked variables.

C. Experiments and Discussion

Experiments were conducted using training, development and test set. Training set was used to train model, while development set was used for validating the model and tuning hyper parameters. Results are then reported for test set. Adam [37] was used as an optimizer for both the tasks with learning rate as 0.0001 and decay of 10^{-6}. Along with dropout, L2 regularization was used as weight regularize to avoid overfitting of the model with a radius as 0.0001.

Table VI shows results on SemEval dataset. We have considered [38] as baseline for this task which is available in Standford core NLP suite. Results show that Bidirectional LSTMs on a word level with dropout and high forget bias initializer is able to beat baseline. Adding attention to this improves the results marginally. Also, we have considered method proposed in [39] as baseline where they have used 1-Dimension CNN followed by max-pooling layer. 1-D CNN or temporal convnets were proven effective for handwritten text classification, evaluated over Amazon review dataset. For baseline, we have used two layer 1-Dimension CNN with the number of filters as 10 for both layers and filter size as 8 and 10 respectively. Max pooling of stride 2 was also applied followed by dropout of 0.2. Incorporating character level CNN does not provide significant improvement as can be seen from the results. Our proposed model which uses word level and character level BiLSTMs and attention over word level BiLSTMs provide a significant increase in the performance.

Results on WASSA datasets are presented in Table V. This task presented a baseline which used embeddings and lexicons as features and SVM as the final regression model. As shown in the table, LSTMs and CNN were unable to beat the baseline due to lack of features. BiLSTMs with dropout and forget bias initializers provide a significant boost in the performance. Attention provides a marginal improvement in

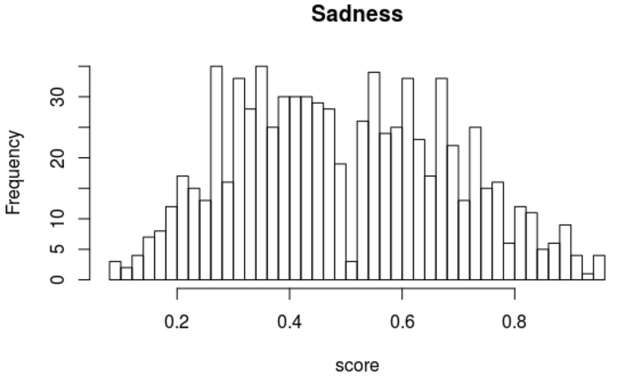

Fig. 7. Distribution of Intensity values for Sadness Emotion

TABLE V
RESULTS ON WASSA DATASET

Model	Average Pearson	Average Spearman	Anger		Fear		Joy		Sadness	
			Per.	Spr.	Per.	Spr.	Per.	Spr.	Per.	Spr.
baseline(task)	0.648	0.641	0.639	0.615	0.652	0.635	0.654	0.662	0.648	0.651
CNN	0.384	0.382	0.237	0.255	0.364	0.361	0.391	0.396	0.544	0.516
LSTM	0.633	0.622	0.579	0.569	0.652	0.636	0.637	0.635	0.664	0.648
BiLSTM	0.631	0.618	0.584	0.569	0.672	0.655	0.618	0.617	0.652	0.632
BiLSTM + Dropout + Bias Initilizers	0.669	0.655	0.630	0.606	0.709	0.695	0.651	0.648	0.687	0.673
BiLSTM + Attention	0.679	0.671	0.641	0.629	0.712	0.697	0.660	0.659	0.702	0.698
BiLSTM + Attention + CNN(char)	0.676	0.659	0.640	0.610	0.705	0.693	0.667	0.650	0.692	0.681
Proposed Model	**0.701**	**0.688**	**0.672**	**0.650**	**0.722**	**0.702**	**0.699**	**0.681**	**0.730**	**0.718**

TABLE VI
RESULTS ON SEMEVAL DATASET

Models	MAE^M	MAE
Stanford (Baseline)	1.13	0.95
BiLSTM(word)	0.9971	0.5547
BiLSTM(word) + Attention	0.9876	0.5538
BiLSTM(word) + BiLSTM(char)	0.9788	0.5569
BiLSTM(word) + CNN(char)	0.9823	**0.5387**
Proposed Model	**0.9175**	0.5541

the performance, although when it is combined with character level BiLSTMs significant improvement is observed.

V. CONCLUSION

In this paper, robust representation of short text is presented which can efficiently predict or classify fine grained emotions. Model presented was evaluated over two datasets and it has shown significant improvement in the results without taking into account any hand crafted features. This work can be expanded to more datasets and can be made more robust if transfer learning is incorporated over different datasets.

REFERENCES

[1] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[2] C. Akkaya, J. Wiebe, and R. Mihalcea, "Subjectivity word sense disambiguation," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*. Association for Computational Linguistics, 2009, pp. 190–199.

[3] S. Poria, E. Cambria, G. Winterstein, and G.-B. Huang, "Sentic patterns: Dependency-based rules for concept-level sentiment analysis," *Knowledge-Based Systems*, vol. 69, pp. 45–63, 2014.

[4] R. Sharma and P. Bhattacharyya, "Detecting domain dedicated polar words." in *IJCNLP*, 2013, pp. 661–666.

[5] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[6] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[7] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[8] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for twitter sentiment classification." in *ACL (1)*, 2014, pp. 1555–1565.

[9] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[10] R. Socher, E. H. Huang, J. Pennin, C. D. Manning, and A. Y. Ng, "Dynamic pooling and unfolding recursive autoencoders for paraphrase detection," in *Advances in Neural Information Processing Systems*, 2011, pp. 801–809.

[11] J. Mitchell and M. Lapata, "Composition in distributional models of semantics," *Cognitive science*, vol. 34, no. 8, pp. 1388–1429, 2010.

[12] J. Deriu, M. Gonzenbach, F. Uzdilli, A. Lucchi, V. De Luca, and M. Jaggi, "Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision." in *SemEval@ NAACL-HLT*, 2016, pp. 1124–1128.

[13] F. Kunneman, C. Liebrecht, and A. van den Bosch, "The (un) predictability of emotional hashtags in twitter," 2014.

[14] S. M. Mohammad, S. Kiritchenko, and X. Zhu, "Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets," *arXiv preprint arXiv:1308.6242*, 2013.

[15] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity in phrase-level sentiment analysis," in *Proceedings of the conference on human language technology and empirical methods in natural language processing*. Association for Computational Linguistics, 2005, pp. 347–354.

[16] X. Ding, B. Liu, and P. S. Yu, "A holistic lexicon-based approach to opinion mining," in *Proceedings of the 2008 international conference on web search and data mining*. ACM, 2008, pp. 231–240.

[17] F. Å. Nielsen, "A new anew: Evaluation of a word list for sentiment analysis in microblogs," *arXiv preprint arXiv:1103.2903*, 2011.

[18] S. Kiritchenko, X. Zhu, and S. M. Mohammad, "Sentiment analysis of short informal texts," *Journal of Artificial Intelligence Research*, vol. 50, pp. 723–762, 2014.

[19] S. M. Mohammad and P. D. Turney, "Crowdsourcing a word–emotion association lexicon," *Computational Intelligence*, vol. 29, no. 3, pp. 436–465, 2013.

[20] F. Bravo-Marquez, E. Frank, S. M. Mohammad, and B. Pfahringer, "Determining word–emotion associations from tweets by multi-label classification," in *WI'16*. IEEE Computer Society, 2016, pp. 536–539.

[21] A. Esuli and F. Sebastiani, "Sentiwordnet: A high-coverage lexical resource for opinion mining," *Evaluation*, pp. 1–26, 2007.

[22] R. J. Williams and D. Zipser, "Backpropagation," Y. Chauvin and D. E. Rumelhart, Eds. Hillsdale, NJ, USA: L. Erlbaum Associates Inc., 1995, ch. Gradient-based Learning Algorithms for Recurrent Networks and Their Computational Complexity, pp. 433–486.

[23] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Neurocomputing: Foundations of research," J. A. Anderson and E. Rosenfeld, Eds.

Cambridge, MA, USA: MIT Press, 1988, ch. Learning Representations by Back-propagating Errors, pp. 696–699.

[24] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[26] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *Trans. Neur. Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[27] A. Graves, "Generating Sequences With Recurrent Neural Networks," *ArXiv e-prints*, Aug. 2013.

[28] J. Henderson, "Discriminative training of a neural network statistical parser," in *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2004, p. 95.

[29] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding." in *Interspeech*, 2013, pp. 3771–3775.

[30] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[31] J. Martens and I. Sutskever, "Learning recurrent neural networks with hessian-free optimization," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 1033–1040.

[32] F. A. Gers, J. A. Schmidhuber, and F. A. Cummins, "Learning to forget: Continual prediction with lstm," *Neural Comput.*, vol. 12, no. 10, pp. 2451–2471, Oct. 2000.

[33] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 2342–2350.

[34] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[35] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-Normalizing Neural Networks," *ArXiv e-prints*, Jun. 2017.

[36] S. M. Mohammad and F. Bravo-Marquez, "WASSA-2017 shared task on emotion intensity," in *Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA)*, Copenhagen, Denmark, 2017.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[38] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, C. Potts *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, vol. 1631, 2013, p. 1642.

[39] X. Zhang and Y. LeCun, "Text Understanding from Scratch," *ArXiv e-prints*, Feb. 2015.